Efficient Nonparametric Belief Propagation with Application to Articulated Body Tracking

Tony X. Han Huazhong Ning Thomas S. Huang
Beckman Institute and ECE Department
University of Illinois at Urbana-Champaign
405 N. Mathews Ave., Urbana, IL 61801
{xuhan, hning2, huang}@ifp.uiuc.edu
http://www.ifp.uiuc.edu/∼xuhan

Abstract

*An efficient Nonparametric Belief Propagation (NBP) algorithm is developed in this paper. While the recently proposed nonparametric belief propagation algorithm has wide applications such as articulated tracking [22, 19], superresolution [6], stereo vision and sensor calibration [10], the hardcore of the algorithm requires repeatedly sampling from products of mixture of Gaussians, which makes the algorithm computationally very expensive. To avoid the slow sampling process, we applied mixture Gaussian density approximation by mode propagation and kernel fitting [2, 7]. The products of mixture of Gaussians are approximated accurately by just a few mode propagation and kernel fitting steps, while the sampling method (*e.g. *Gibbs sampler) needs many samples to achieve similar approximation results. The proposed algorithm is then applied to articulated body tracking for several scenarios. The experimental results show the robustness and the efficiency of the proposed algorithm. The proposed efficient NBP algorithm also has potentials in other applications mentioned above.*

1. Introduction

Many computer vision problems involve estimation of statical properties of random variables. These random variables are usually high-dimensional, continuous and multimodal distributed, which partially explains why most of the computer vision problems are so challenging. Graphical model such as Bayesian network, Markov Random Field (MRF) and Conditional Random Field (CRF) represent statistical dependencies of random variables by a graph. Learning and inference of the high dimensional random variables therefore become much easier. Belief Propagation (BP) [15, 12] is a powerful and elegant inference algorithm for graphical model. BP can achieve exact inference for acyclic graph. The exact inference for general graph is NP hard. However, simply applying BP on the loopy graph gives satisfied results [5, 26, 14].

For many vision problems, the random variables of each nodes in the graphical model are continuous and high dimensional (*e.g.* articulated body tracking). The BP algorithm for this kind of model involves integral equations for which close form solution seldom exists. The straightforward thought would be discretization. Unfortunately, for high-dimensional case, the exhaustive discretization of the state space is infeasible. To tackle this difficulty, the Nonparametric Belief Propagation (NBP) is proposed [21, 11]. Instead of discretizing the entire high-dimensional state space, NBP uses mixture of Gaussians to approximate the continuous potential functions of the graph. For each iteration, the parameters of the mixture of Gaussians are recomputed using Gibbs sampling. The computational complexity for each node is $O(d\kappa M^2)$, where d is the degree of the node, M the number of samples and κ the fixed iteration number of the Gibbs sampler. To ensure good approximation, the Gibbs sampler require a large number of particles (a typical setting is $M = 100$ and $\kappa = 100$ in [21], which make the NBP algorithm inevitably slow. According to [22], with 200 particles, the matlab implementation requires about one minute for each NBP iterations.

Although NBP has been a popular inference algorithm for graphical model with high dimensional node [22, 19, 6, 10], the formidable computation complexity restricts its application in many scenarios, where running time is critical.

To avoid the slow sampling based technique, we applied mixture Gaussian density approximation based on variable-bandwidth mean-shift [2] and kernel fitting [7]. The products of mixture of Gaussians are approximated accurately by just a few mode propagation and kernel fitting steps. While the sampling method (*e.g.* Gibbs sampler) needs lots

Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)
0-7695-2597-0/06 $20.00 © 2006 IEEE

of samples to achieve similar approximation results.

We then applied the proposed algorithm to articulated body tracking, which is a very challenging problem and a discriminating test bed for inference machinery. The experimental results for several scenarios show the robustness and the efficiency of the proposed algorithm.

The main contribution of this paper lies in 1) Improve the efficiency of the Nonparametric Belief Propagation (NBP) by sequentially mode propagation and kernel fitting; 2) Demonstrating the efficiency and robustness of the improved NBP by implementing an articulated body tracker, which achieved satisfied results in several scenarios.

1.1. Related Work on Articulated Body Tracking

Articulated human body tracking is inherently a very difficult problem due to: 1) high degree (usually 20–68) of freedom of the articulated body movement [20, 4, 16, 1]; 2) large appearance change of body parts during the movement; 3) occlusion between body parts; 4) no typical appearance due to clothing; 5) fast movement of human arms and legs; 6) the posterior distribution of body configuration is multimodal and spiky.

Many enlightening articulated body tracking algorithms appeared in recent years. Bregler and Malik used exponential map to model the articulated twist. After model the kinematic chain as the product of exponentials, a Newton-Raphson style minimization is carried out to find the minimizer (the body configuration) of the cost function [1]. The introduction of the exponential map is a neat idea, but the Newton-Raphson is inherently a variant of the gradient descent method. Therefore, the optimization procedure is likely to be trapped by local minima. We have mentioned above that the posterior distribution of body configuration is multimodal and spiky. In other words, there are many local minima in the objective function which usually make the local minimizer found by the tracker different from the global minimizer, i.e. the optimal body configuration given the current observation. The fact that body parts usually move very fast compared with the common frame rate, further validates the claim. Annealing the particle filter may be one way to tackle this difficulty [3]. However, the sampling based algorithms are usually very slow when large amount of particles are required. The high dimensionality of articulated body motion requires large number of particles even if the annealed particle filter is applied. According to [3], the tracker using 10 annealing layers with 200 particles need around 1 hour to process 5 seconds of footage.

Ramanan and Forsyth proposed a 2d tracker with automatic initialization, which can track long video sequence [16, 17]. The continuous body configuration space is discretized first and a variant of BP, max product, is then carried out on the loopy graphical model to find the approximate estimate of the MAP, i.e. the suboptimal body configuration given current video input. BP reduces the computational complexity of brutal force search from $O(N^m)$ to $O(N^2m)$ for non-loopy graph. Here N is the size of the domain of each node (the possible discrete values the node can take) and m is the number of nodes in the graphical model. But usually N, i.e. the domain size of each node, is very large in the tracking context. For articulated 2d tracker, the configuration of each body part is a triplet $(x, y, \theta)^T$, representing the horizontal translation, vertical translation, and in-plane rotation respectively. Suppose each axis is discretized into 20 bins, the searching space for each node is 20^3. As we have pointed out, the computation needed for the the 2d tracker is therefore $20^6 \cdot 9 \cdot C_T$, where C_T is the computation of template comparison for each body part. Consequently, the articulated tracker based on the BP in the discretized space requires huge computational power or lots of cpu time. The complexity would be even formidable if we want to realize a 3d articulated tracker using the same approach.

Sigal *et al*. [19] devised a 3D articulated body tracker and Sudderth *et al*. [22] came up with a articulated hand tracker both based on NBP. Although these two trackers require lots of computation, both of them achieve very impressive results.

The rest of the paper is organized as follows. We first give a brief review on Belief Propagation (BP) and Nonparametric Belief Propagation (NBP) in section 2. Section 3 explained in detail on how to accelerate NBP by efficiently approximating mixture of Gaussians using mode propagation and kernel fitting. The implementation of an articulated body tracker using proposed fast NBP is then described in section 4. Experiment results and the discussion are given in section 5. And section 6 concludes the paper.

2. BP and NBP

An undirected graphical model or a Markov Random Field (MRF) is a graph in which the nodes represent random variables and edges represent compatibility constraints between them. If all probabilities in the graph are nonzero, the Hammersley-Clifford theorem guarantees that the joint probability distribution will factorize into a product of the maximal cliques of the graph [15]. Denote the graph as $\mathcal{G}$, its vertex set as $\mathcal{V}$ and its edge set as $\mathcal{E}$. The neighborhood of a node $s \in \mathcal{V}$ is defined as $\Gamma(s) \triangleq \{t|(s,t) \in \mathcal{E}\}$. Each node $s \in \mathcal{V}$ is associated with a hidden random variable $\mathbf{x}_s$. The noisy local observation of $\mathbf{x}_s$ is denoted as $\mathbf{y}_s$. $x = \{\mathbf{x}_s|s \in \mathcal{V}\}$ and $y = \{\mathbf{y}_s|s \in \mathcal{V}\}$ denote the set of all hidden and observed variables, respectively. To make the introduction of BP succinct, we only consider the model with pairwise compatibility functions (The BP and NBP can be directly extended to graphical models with clique ≥ 3). Therefore the joint distribution $p(x, y)$ can be factorized as:

$$p(x,y) = \prod_{(s,t)\in\mathcal{E}} \Psi_{s,t}(\mathbf{x}_s,\mathbf{x}_t) \prod_{s\in\mathcal{V}} \Phi_s(\mathbf{x}_s,\mathbf{y}_s)\ . \quad (1)$$

where $\mathcal{E}$ and $\mathcal{V}$ are the nodes set and the edges set of the graph, respectively.

2.1. BP

For inference on the graphical model, we are interested in calculating the conditional marginal distribution $p(\mathbf{x}_s|y)$. For acyclic graph, $p(\mathbf{x}_s|y)$ can be exactly computed by BP, which also achieves satisfied approximation for loopy graph [5, 26, 14].

At the nth iteration of BP, each node $t \in \mathcal{V}$ send a message $m_{ts}^n(\mathbf{x}_s)$ to its neighbor $s \in \Gamma(t)$:

$$m_{ts}^n(\mathbf{x}_s) = \int_{\mathbf{x}_t} \Psi_{s,t}(\mathbf{x}_s,\mathbf{x}_t)\Phi_t(\mathbf{x}_t,\mathbf{y}_t) \times \prod_{u\in\Gamma(t)\setminus s} m_{ut}^{n-1}(\mathbf{x}_t)d\mathbf{x}_t \quad (2)$$

At any iteration of BP, for each node, the approximation of the true conditional marginal $p(\mathbf{x}_s|y)$ can be computed by combining the incoming messages from the neighborhood:

$$\hat{p}^n(x,y) = \alpha\Phi_s(\mathbf{x}_s,\mathbf{y}_s) \prod_{t\in\Gamma(s)} m_{ts}^n(\mathbf{x}_s)\ , \quad (3)$$

where α is a normalization constant.

For acyclic graph $\mathcal{G}$, the approximate conditional marginal, $\hat{p}^n(\mathbf{x}_s|y)$ will converge to the true conditional marginal $p(\mathbf{x}_s|y)$ after d iteration, where d is the diameter of $\mathcal{G}$.

2.2. NBP

For continuous random variable, it is usually infeasible to analytically evaluate the BP message update in equation 2. Although we can always approximate the analytical results by discretization, for high-dimensional case (*e.g.* articulated tracking), the exhaustive discretization of the state space is also impractical. To tackle this difficulty, Sudderth *et al*. [21] and Isard [11] proposed the Nonparametric Belief Propagation (NBP). Instead of discretized the entire high-dimensional state space, the messages $m_{ts}(\mathbf{x}_s)$ are represented nonparametrically as a kernel density estimate shown below.

$$m_{ts}(\mathbf{x}_s) = \sum_{i=1}^{M} \omega_s^{(i)} \mathcal{N}\left(\mathbf{x}_s;\mu_s^{(i)},\mathbf{\Lambda}_s\right)\ , \quad (4)$$

where $\mathcal{N}(x;\mu,\mathbf{\Lambda})$ denote Gaussian distribution with mean μ and covariance matrix $\mathbf{\Lambda}$. $\omega_s^{(i)}$s are mixture weights with the constraints $\sum_{i=1}^{M}\omega_s^{(i)} = 1$. By assuming all compatibility functions are mixture of Gaussians, the propagating messages in equation 2 are also mixture of Gaussians. This is because that the product of d Gaussian densities is itself Gaussian, with mean and covariance matrix give by

$$\prod_{j=1}^{d} \mathcal{N}(\mathbf{x};\mu_j,\mathbf{\Lambda}_j) \propto \mathcal{N}(\mathbf{x};\bar{\mu},\bar{\mathbf{\Lambda}}) \quad (5)$$

$$\bar{\mathbf{\Lambda}}^{-1} = \sum_{j=1}^{d} \mathbf{\Lambda}_j^{-1} \qquad \bar{\mathbf{\Lambda}}^{-1}\bar{\mu} = \sum_{j=1}^{d} \mathbf{\Lambda}_j^{-1}\mu_j \quad (6)$$

Suppose a node has d neighbors and each incoming message from neighbors is a Gaussian mixture of M component. The message passing is operated by multiplying d Gaussian mixtures of M component. This will produce a Gaussian mixture of M^d components. The weights $\bar{\omega}$ associated with product mixture component $\mathcal{N}(x;\bar{\mu},\bar{\mathbf{\Lambda}})$ is

$$\bar{\omega} \propto \frac{\prod_{j=1}^{d}\omega_j\mathcal{N}(\mathbf{x};\mu_j,\mathbf{\Lambda}_j)}{\mathcal{N}(\mathbf{x};\bar{\mu},\bar{\mathbf{\Lambda}})}\ , \quad (7)$$

where $\{\omega_j\}_{j=1}^{d}$ are the weights associated with the input Gaussians. Integrating above mixture of Gaussians in equation 2 is mathematically trivial. In practice, however, the components of the product of mixture of Gaussians increase exponentially with the iteration number.

Therefore a key issue for NBP is **how to accurately approximate the products of mixture of Gaussians while keep the component number in a acceptable range.** In [21], Gibbs sampler is applied to approximate the products of the mixture of Gaussians. Each NBP message update involves two stages: sampling from the estimated marginal, followed by Monte Carlo approximation of the outgoing message. With $M = 200$ particles, the NBP achieves satisfied approximations results. One flaw of the above NBP algorithm is that it is computationally expensive: each NBP iteration requires 1 minute on a Pentium IV workstation [22].

We will show how to efficiently approximate the products of mixture of Gaussians using density estimation by mode propagation and kernel fitting [2, 7] in next section.

3. Accelerating the NBP by Sequential Density Estimation through Mode Propagation

Given d input mixtures of M Gaussians, the resulting product is also mixture of Gaussians of M^d component as shown in equation (5– 7). Denote the resulting mixture of Gaussians as:

$$\hat{f}(\mathbf{x}) = (2\pi)^{-d/2}\sum_{i=1}^{N}\bar{\omega}_i|\bar{\mathbf{\Lambda}}_i|^{-1/2}\exp\left(-\frac{1}{2}D^2(\mathbf{x},\bar{\mu}_i,\bar{\mathbf{\Lambda}}_i)\right), \quad (8)$$

where

$$D^2(\mathbf{x}, \bar{\mu}_i, \bar{\mathbf{\Lambda}}_i) \equiv (\mathbf{x} - \bar{\mu}_i)^T \bar{\mathbf{\Lambda}}_i^{-1} (\mathbf{x} - \bar{\mu}_i) \tag{9}$$

is the Mahalanobis distance from $\mathbf{x}$ to $\bar{\mu}_i$ and $N = M^d$.

Our goal is to efficiently obtain the compact representation of the mixture of Gaussian of M^d component (i.e. the product of d mixtures of Gaussians of M component). The key idea is to merge the "indistinguishable" component by mode detection. The mode location and its weight are found by mean-shift algorithm [2] and the covariance matrix associated with each mode is derived from the Hessian estimated at the mode location.

In order to locate the mode of the mixture of Gaussians, the variable-bandwidth mean shift vector is defined by

$$\mathbf{m}(\mathbf{x}) \triangleq \left(\sum_{i=1}^{N} \kappa_i(\mathbf{x}) \bar{\mathbf{\Lambda}}_i^{-1} \right)^{-1} \left(\sum_{i=1}^{N} \kappa_i(\mathbf{x}) \bar{\mathbf{\Lambda}}_i^{-1} \bar{\mu}_i \right) - \mathbf{x} \tag{10}$$

where the weights

$$\kappa_i(\mathbf{x}) = \frac{\bar{\omega}_i |\bar{\mathbf{\Lambda}}_i|^{-1/2} \exp\left(-\frac{1}{2} D^2(\mathbf{x}, \bar{\mu}_i, \bar{\mathbf{\Lambda}}_i)\right)}{\sum_{i=1}^{n} \bar{\omega}_i |\bar{\mathbf{\Lambda}}_i|^{-1/2} \exp\left(-\frac{1}{2} D^2(\mathbf{x}, \bar{\mu}_i, \bar{\mathbf{\Lambda}}_i)\right)} \tag{11}$$

satisfy $\sum_{i=1}^{N} \kappa_i(\mathbf{x}) = 1$.

It can be shown [2] that by iteratively computing the mean shift (10) and translating the location $\mathbf{x}$ by $\mathbf{m}(\mathbf{x})$, a mode seeking algorithm is obtained, which converges to a stationary point of the mixture of Gaussians (8). Since the maxima of the density are the only stable points of the iterative procedure, most of the time the convergence happens at a mode of (8). A formal check for the localization of the mode involves the computation of the Hessian matrix

$$\begin{aligned} \hat{\mathbf{Q}}(\mathbf{x}) &\triangleq (\nabla \nabla^T) \hat{f}(\mathbf{x}) \\ &= (2\pi)^{-d/2} \sum_{i=1}^{N} \bar{\omega}_i |\bar{\mathbf{\Lambda}}_i|^{-1/2} \exp\left(-\frac{1}{2} D^2(\mathbf{x}, \bar{\mu}_i, \bar{\mathbf{\Lambda}}_i)\right) \cdot \\ &\quad \bar{\mathbf{\Lambda}}_i^{-1} \left((\mathbf{x} - \bar{\mu}_i)(\mathbf{x} - \bar{\mu}_i)^T - \bar{\mathbf{\Lambda}}_i \right) \bar{\mathbf{\Lambda}}_i^{-1}, \end{aligned} \tag{12}$$

which should be negative definite. If it is not negative definite, the converge point is a saddle point or inflection point. In this case, kernels associated with such modes should be restored and considered as separate modes for further processing.

The desired compact approximate density is obtained by detecting the mode locations initiated from every $\bar{\mu}_i$, i.e. the mean of each component of $\hat{f}(\mathbf{x})$. Suppose that the approximate density has N' unique modes denoted as $\tilde{\mu}_j$ ($j = 1 \ldots N'$). The corresponding mix weights $\tilde{\omega}_j$ is equal to the sum of the kernel weights converged to $\tilde{\mu}_j$. For each new mode located at $\tilde{\mu}_j$, we need to estimate its covariance matrix $\tilde{\mathbf{\Lambda}}_j$). According to equation (12), the Hessian matrix at $\tilde{\mu}_j$ is approximated as:

$$-\hat{\mathbf{Q}}(\tilde{\mu}_j) \approx (2\pi)^{-d/2} \tilde{\omega}_j |\tilde{\mathbf{\Lambda}}_j|^{-1/2} \tilde{\mathbf{\Lambda}}_j^{-1} \tag{13}$$

When $\hat{\mathbf{Q}}(\tilde{\mu}_j)$ is negative definite, take the determinant of the both sides of equation (13). By noting the fact that for a matrix $A \in R^{d \times d}$ and a scalar $s \in R$, $|sA| = |s|^d |A|$, $|\tilde{\mathbf{\Lambda}}_j|^{-1/2}$ is estimated as

$$|\check{\mathbf{\Lambda}}_j|^{-1/2} = (2\pi)^{\frac{d}{2} \cdot \frac{d}{d+2}} \tilde{\omega}_j^{-\frac{d}{d+2}} \left| -\hat{\mathbf{Q}}(\tilde{\mu}_j)^{-1} \right|^{-\frac{1}{d+2}}. \tag{14}$$

Substitute $|\tilde{\mathbf{\Lambda}}_j|^{-1/2}$ by $|\check{\mathbf{\Lambda}}_j|^{-1/2}$ in equation (13) we compute the estimation of $\tilde{\mathbf{\Lambda}}_j$) as

$$\check{\mathbf{\Lambda}}_j = \tilde{\omega}_j^{\frac{2}{d+2}} \left| 2\pi \left(-\hat{\mathbf{Q}}(\tilde{\mu}_j)^{-1} \right) \right|^{-\frac{1}{d+2}} \left(-\hat{\mathbf{Q}}(\tilde{\mu}_j)^{-1} \right). \tag{15}$$

The final compact density approximation of the products of mixture of Gaussians is given by

$$\tilde{f}(\mathbf{x}) = (2\pi)^{-d/2} \sum_{i=1}^{N'} \tilde{\omega}_i |\check{\mathbf{\Lambda}}_i|^{-1/2} \exp\left(-\frac{1}{2} D^2(\mathbf{x}, \tilde{\mu}_i, \check{\mathbf{\Lambda}}_i) \right), \tag{16}$$

and $N' \ll N$ is satisfied in most of cases.

3.1. Sequentially Density Approximation by Mode Propagation

By applying above mode propagation and kernel fitting procedure, we can achieve a much more compact ($N' \ll N$) approximation of $\hat{f}(\mathbf{x})$, i.e. the products of the messages represented as mixture of Gaussians.

The density approximation technique described above is accurate and memory efficient, but the computational complexity of the mode detection algorithm for N component is $O(N^2)$. If each message is originally represented by the mixture of many Gaussian components (i.e. M is large), even the degree of the node (i.e. d) in the graphical model is small, the computational complexity will be $O(M^{2d})$, which make the mean-shift based mode detection algorithm very time consuming (although faster than the sampling based Gibbs sampler).

Therefore the sequential density approximation is applied. Instead of approximating the products of d messages in one step, the sequential density approximation achieves the compact Gaussian mixture representation by $d - 1$ step. That is, first multiply two messages represented as the Gaussians mixtures of M components and apply above density approximation algorithm. The approximated compact representation $\tilde{f}(\mathbf{x})$ is then multiplied with the third

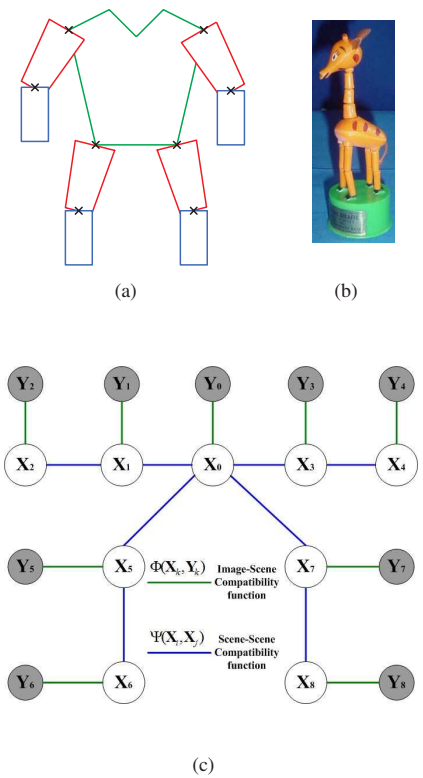
nonparametric messages. The density approximation is applied again. The above procedure is repeated until the dth nonparametric message is multiplied and approximated.

The complexity of sequential density approximation is $O\left(M^4(d-1)\right)$, which is quite acceptable. Our articulated body tracker based on the NBP accelerated by the sequential density approximation, can achieve 1.5 frames/second for 1024×768 video.

4. Applying Efficient NBP to Articulated Body Tracking

4.1. Graphical Body Model with Elastic Constraints

We model the 2D view of the human body as a connected card board model shown in figure 1(a). In our body tracking framework, we use a "loose-limbed" body model [20, 4] in which the limbs are not rigidly connected but are rather "attracted" to each other. Instead of representing the body as a single 33-dimensional kinematic tree, each limb is treated quasi-independently with soft constraints between the position and orientation of adjacent parts. The model resembles a Push Puppet toy which has elastic connections between the limbs as shown in figure 1(b). For each body part, i.e. the node in the graph shown in figure 1(c), the configuration space is a 3 dimensional vector space (x, y, θ), representing the horizontal translation, vertical translation, and in-plane rotation respectively. The elastic constraints between body parts are modeled as the Scene-Scene compatibility function $\Psi(\mathbf{X}_i, \mathbf{X}_j)$. The closer the two nodes in spacial distance, the bigger the scene-scene compatibility function. The Image-Scene compatibility function $\Phi(\mathbf{X}_k, \mathbf{Y}_k)$ model the similarity between the tracked parts and the parts' template.



(a) (b) (c)

Figure 1. The model setup for the 2D articulated body tracker.
(a) The card board model of human body with elastic joints.
(b) Toy Push Puppet with elastic joints.
(c) The loose attractive graphical body model, on which the belief propagation is carried on. $\mathbf{X}_0$: Torso configuration; $\mathbf{X}_1$: Right upper arm configuration; $\mathbf{X}_2$: Right forearm configuration; $\mathbf{X}_3$: Left upper arm configuration; $\mathbf{X}_4$: Left forearm configuration; $\mathbf{X}_5$: Right thigh configuration; $\mathbf{X}_6$: Right crus configuration; $\mathbf{X}_7$: Left thigh configuration; $\mathbf{X}_8$: Left crus configuration; $\mathbf{Y}_i$ is the observation of $\mathbf{X}_i$. The random variable $\mathbf{X}_i$'s only directly depend on their neighbors.

4.2. Substantiating the Compatibility Functions

In our implementation of the tracker, the template of each body part is manually labeled at the first frame of the video, or, at a typical frame where no self-occlusion exists. The template for each body part is actually an image patch (containing both edge and texture) encompassed by a polygon as shown in figure 2. Denote the template for jth body part as T_j and the warped version of the template $T_j\langle\mathbf{X}_j\rangle$ according to the configuration parameter $X_j = (x, y, \theta)^T$. Denote the tth frame of the tracking video as I_t and the observed image patch for jth body part with configuration $\mathbf{X}_j$ is therefore $I_j\langle\mathbf{X}_j\rangle$. The image-scene compatibility function $\Phi(\mathbf{X}_k, \mathbf{Y}_k)$, which is proportional to the conditional probability $P(\mathbf{Y}_k|\mathbf{X}_k)$ is therefore defined as

$$\begin{aligned}\Phi(\mathbf{X}_k, \mathbf{Y}_k) &\propto P(\mathbf{Y}_k|\mathbf{X}_k) \\ &= C_k \exp\left(\alpha SSD_{rgb}\left(I_j\langle\mathbf{X}_j\rangle, T_k\langle\mathbf{X}_k\rangle\right)\right) \\ &\times \exp\left((1-\alpha)Chamfer\left(I_j\langle\mathbf{X}_j\rangle, T_k\langle\mathbf{X}_k\rangle\right)\right) , \quad (17)\end{aligned}$$

where $SSD_{rgb}(\cdot,\cdot)$ is the Sum of Squared Difference (SSD) between image patch and template in RGB color space. $Chamfer(\cdot,\cdot)$ is the chamfer distance between the edge point sets of template and image patch [23]. α is a predefined parameter to balance the observation preference between appearance and edge. C_k is the normalization constant. The chamfer distance function is efficiently computed using a distance transform (DT) [23].

The elastic constraints between body parts, i.e. the scene-scene compatibility function $\Psi(\mathbf{X}_i, \mathbf{X}_j)$ is character-

ized by the Euclidean distance between the joint points of the adjacent body parts. For two adjacent body parts i and j, denote the soft joint on ith part, which is elastically connected to the jth part, as $\mathbf{J}_{i\to j}$. The corresponding soft joint on ith part is $\mathbf{J}_{j\to i}$. The 2D location of $\mathbf{J}_{i\to j}$ with ith body part's configuration $\mathbf{X}_i$, is $\mathbf{J}_{i\to j}\langle\mathbf{X}_i\rangle$. Therefore the scene-scene compatibility function is defined as:

$$\Psi(\mathbf{X}_i,\mathbf{X}_j)=\exp\left(K\left\|\mathbf{J}_{i\to j}\langle\mathbf{X}_i\rangle-\mathbf{J}_{j\to i}\langle\mathbf{X}_j\rangle\right\|_2\right), \tag{18}$$

where K is the spring constant depending on how "elastic" we want the body card model to be. In order to adapt the above elastic constrains function in the NBP framework, we use a set (20 component) of circularly symmetric Gaussians to approximate the potential function 18.

With the compatibility functions define above, The efficient NBP can be carried out on the body graphical model as discussed in section 2 and section 3.

5. Experiment Results

We use C++ programming language to realize a 2d articulated body tracker based on efficient NBP and test it in several scenarios including dancing, meeting and treadmill walking. The 2d tracker achieve robust tracking for both frontal view (figure 2 3 4) and oblique view (figure 5). The challenging parts of the video data show the effectiveness of our algorithm. Firstly, the tracker can handle the partial self-occlusion (e.g. in the third image in Figure 4 where the left upper arm was occluded by the lower arm), thanks to the BP that the tracker can infer from partly missing information. Secondly, the tracker is able to recover quickly from the failures. For example, in figure 5 the lower arms are lost for sometime due to large 3D motion, however as soon as the subject takes a pose not very different from the initial pose, the tracker can recover true lower arm positions by inference from the torso and upper arm positions. Thirdly, the combination of both appearance and edge information solves many difficulties which will occur if only single modality is used. One example is the 7^{th} and 8^{th} images in Figure 2 where the two legs lacking texture are near each other. In this situation, the tracker without edge modality will give a result of leg overlapping.

Another important improvement of the proposed NBP is that it accelerates the tracker a lot. For each node, the size of all searching space is $20^3=8000$. Therefore, if we directly apply BP, the computation needed for the the 2d tracker is $20^6\cdot 9\cdot C_T$, where C_T is the computation of template comparison for each body part. While the efficient NBP based articulated 2d tracker can achieve 1.5 frames/second in average for the shown 1024×768 video on a Pentium IV 2.4G desktop, which is around 80 times faster than our implementation of the tracker based on BP.

The tracker may fail in some special situations. For the oblique view, at some frames the tracker lost the track of the lower arms. This is due to the severe self-occlusion and large 3D motion. The 2D articulated tracker cannot handle the above 2 situations, therefore an articulated 3d tracker is our future work. In that case, the searching space exponentially grows and a faster algorithm is indispensable.

6. Conclusion/Discussion

An efficient Nonparametric Belief Propagation (NBP) algorithm is proposed in this paper. A 2d articulated body tracker based on the efficient NBP achieved satisfied results for several scenario, including dancing, meeting and treadmill walking. However, our current 2d articulated tracker cannot handle severe self occlusion and singularity caused by 3d movement [8]. Therefore implementing a 3D articulated body tracker based on the efficient NBP algorithm is our future work. We believe the increasing dimensionality of node in the graphical model (change from (x,y,θ) to (x,y,z,α,β)) provides a good chance to demonstrate the strength of the proposed efficient NBP algorithm.

References

[1] C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *CVPR*, page 8, 1998.

[2] D. Comaniciu, V. Ramesh, and P. Meer. The variable bandwidth mean shift and data-driven scale selection. In *ICCV*, pages 438–445, 2001.

[3] J. Deutscher, A. Blake, and I. Reid. Articulated body motion capture by annealed particle filtering. In *CVPR00*, pages II: 126–133, 2000.

[4] P. Felzenszwalb and D. Huttenlocher. Efficient matching of pictorial structures. In *CVPR*, pages 66–73, 2000.

[5] W. Freeman, E. Pasztor, and O. Carmichael. Learning low-level vision. *IJCV*, 40:25–47, 2000.

[6] M. D. Gupta, S. Rajaram, N. Petrovic, and T. S. Huang. Restoration and recognition in a loop. In *CVPR*, pages 638–644, 2005.

[7] B. Han, D. Comaniciu, Y. Zhu, and L. S. Davis. Incremental density approximation and kernel-based bayesian filtering for object tracking. In *CVPR*, pages 638–644, 2004.

[8] T. X. Han and T. S. Huang. Articulated body tracking using dynamic belief propagation. In *ICCV-HCI*, pages 26–35, 2005.

[9] A. T. Ihler, I. John W. Fisher, R. L. Moses, and A. S. Willsky. Nonparametric belief propagation for self-calibration in sensor networks. In *IPSN'04*, pages 225–233, 2004.

[10] A. T. Ihler, E. B. Sudderth, W. T. Freeman, and A. S. Willsky. Efficient multiscale sampling from products of gaussian mixtures. In *NIPS*, 2003.

[11] M. Isard. Pampas: real-valued graphical models for computer vision. In *CVPR*, 2003.

[12] F. Kschischang and B. Frey. Factor graphs and the sum-product algorithm. *IEEE Trans. on Information Theory*, 47:498–519, February 2001.

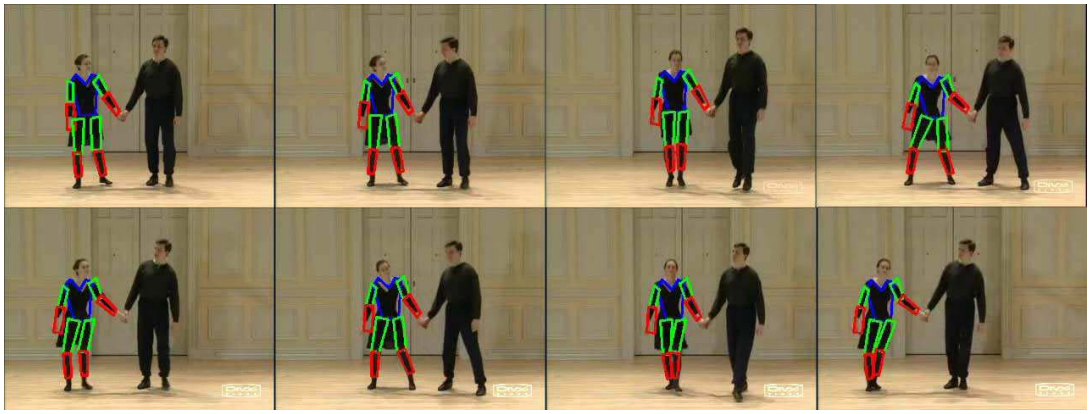


Figure 2. Tracking results of dancing subject. (Video size is 320×240)

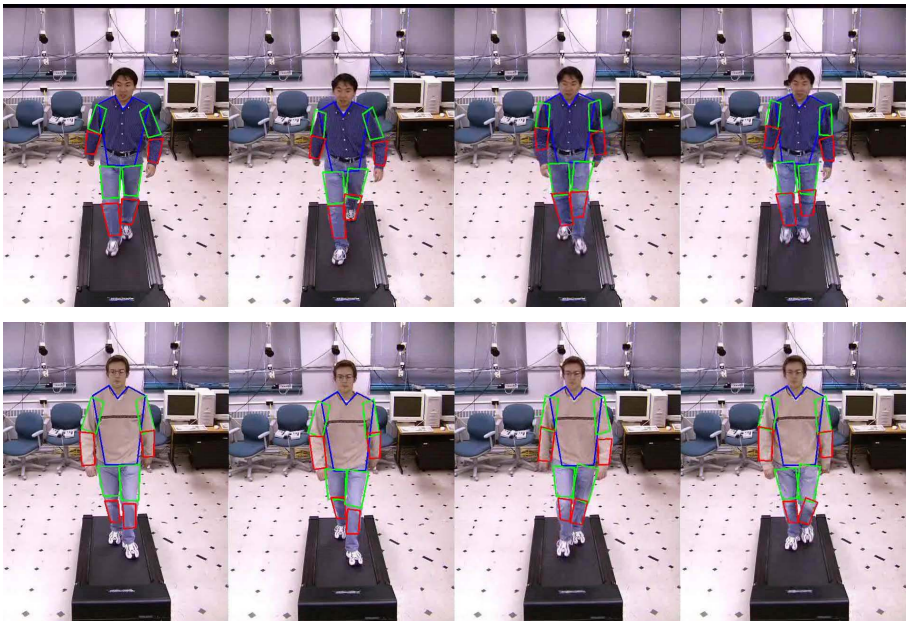


Figure 3. Tracking results of treadmill walking [18]. (Video size is 486×640)

[13] M. Liu, T. X. Han, and T. S. Huang. Online appearance learning by template prediction. In *AVSS*, 2005.

[14] K. P. Murphy, Y. Weiss, and M. I. Jordan. Loopy belief propagation for approximate inference: An empirical study. In *Proc. Uncertainty in AI*, pages 467–475, 1999.

[15] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988.

[16] D. Ramanan and D. A. Forsyth. Finding and tracking people from the bottom up. In *CVPR*, pages 467–474, 2003.

[17] D. Ramanan, D. A. Forsyth, and A. Zisserman. Strike a pose: Tracking people by finding stylized poses. In *CVPR*, 2005.

[18] R.Gross and J. Shi. The cmu motion of body (mobo) database. *Tech. Report CMU-RI-TR-01-18, Robotics Institute, Carnegie Mellon University*, 2001.

[19] L. Sigal, S. Bhatia, S. Roth, M. Black, and M. Isard. Tracking loose-limbed people. In *CVPR04*, pages I: 421–428, 2004.

[20] L. Sigal, M. Isard, B. H. Sigelman, and M. J. Black. Attractive people: Assembling loose-limbed models using non-parametric belief propagation. In *NIPS*, 2003.

[21] E. Sudderth, T. Ihler, W. Freeman, and A. Willsky. Nonparametric belief propagation. *Proc. CVPR*, pages 605–612, 2003.

[22] E. B. Sudderth, M. I. Mandel, W. T. Freeman, and A. S. Willsky. Distributed occlusion reasoning for tracking with

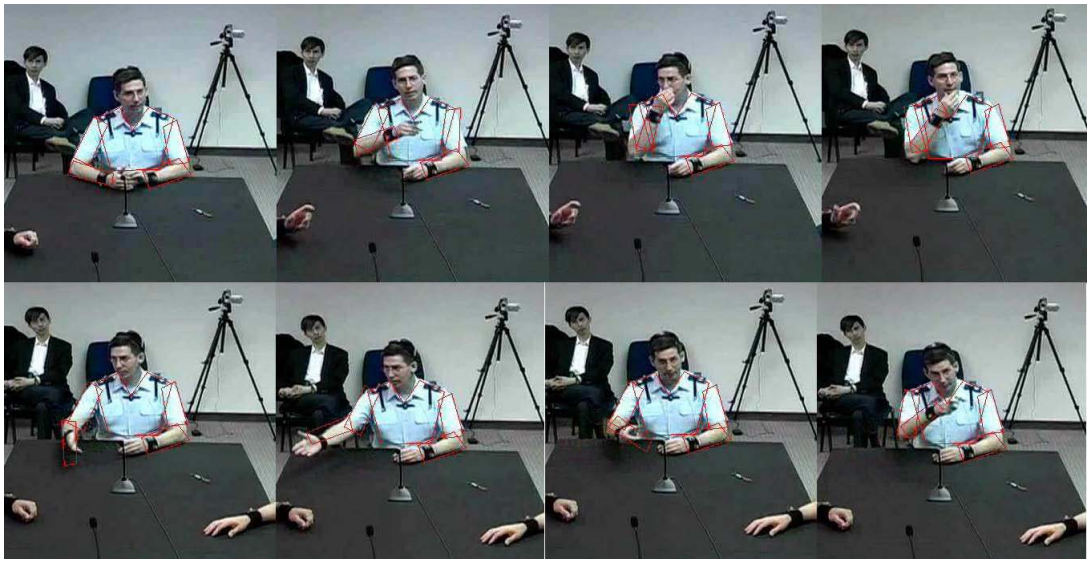


Figure 4. Tracking results of body movement during a meeting (frontal view). Original video size is 1024×768 and the frames shown are cropped version

Figure 5. Tracking results of body movement during a meeting (oblique view). Original video size is 1024×768 and the frames shown are cropped version

nonparametric belief propagation. In *Advances in Neural Information Processing Systems 17*, pages 1369–1376. MIT Press, Cambridge, MA, 2005.

[23] A. Thayananthan, B. Stenger, P. H. S. Torr, and R. Cipolla. Shape context and chamfer matching in cluttered scenes. In *CVPR*, pages 127–133, 2003.

[24] Y. Wu, G. Hua, and T. Yu. Tracking articulated body by dynamic markov network. In *ICCV03*, pages 1094–1101, 2003.

[25] Y. Wu and T. Huang. Hand modeling, analysis and recognition. *IEEE Signal Processing Magazine*, 18:51–60, May 2001.

[26] J. Yedidia, W. Freeman, and Y. Weiss. Understanding belief propagation and its generalization, 2001.